KELOMPOK:

Moh. Naufal Faqih 10222044

Firman Firdaus 10222033

Ryan Azis S. 10222041

---

1. Fitur Seleksi

1.1. Memuat Dataset Diabetes

```
        age       sex       bmi        bp        s1        s2        s3  \
0  0.038076  0.050680  0.061696  0.021872 -0.044223 -0.034821 -0.043401
1 -0.001882 -0.044642 -0.051474 -0.026328 -0.008449 -0.019163  0.074412
2  0.085299  0.050680  0.044451 -0.005670 -0.045599 -0.034194 -0.032356
3 -0.089063 -0.044642 -0.011595 -0.036656  0.012191  0.024991 -0.036038
4  0.005383 -0.044642 -0.036385  0.021872  0.003935  0.015596  0.008142

         s4        s5        s6
0 -0.002592  0.019907 -0.017646
1 -0.039493 -0.068332 -0.092204
2 -0.002592  0.002861 -0.025930
3  0.034309  0.022688 -0.009362
4 -0.002592 -0.031988 -0.046641
0    151.0
1     75.0
2    141.0
3    206.0
4    135.0
Name: target, dtype: float64
```

1.2. Menghitung Korelasi Pearson antara Fitur dengan label

```
target    1.000000
bmi       0.586450
s5        0.565883
bp        0.441482
s4        0.430453
s3        0.394789
s6        0.382483
s1        0.212022
age       0.187889
s2        0.174054
sex       0.043062
Name: target, dtype: float64
```

1.3. Memilih 5 Fitur Terbaik Menggunakan RFE

```
Selected features using RFE:
['bmi', 's5', 'bp', 's4', 's3']
```

1.4. Membuat Model Prediksi & Evaluasi Performa Model

```
Accuracy of the Logistic Regression model: 0.7865168539325843

Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.76      0.80        49
           1       0.73      0.82      0.78        40

    accuracy                           0.79        89
   macro avg       0.79      0.79      0.79        89
weighted avg       0.79      0.79      0.79        89
```

Hasil:

- Konversi label ke bentuk biner membuat model klasifikasi jauh lebih efektif dibandingkan model regresi awal.
- Akurasi model sebesar ~79% menunjukkan bahwa model cukup baik dalam membedakan pasien dengan risiko tinggi dan rendah.
- Precision dan recall seimbang, baik pada kelas 0 maupun kelas 1, menunjukkan bahwa model tidak bias secara signifikan terhadap salah satu kelas.
- Macro dan weighted average f1-score = 0.79 menandakan performa stabil di kedua kelas.

2. Reduksi Dimensi

2.1. Muat MNIST Dataset

```
mnist = fetch_openml('mnist_784', version=1)
X = mnist.data
y = mnist.target

X = X / 255.0

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

2.2. Gunakan PCA untuk Mereduksi Dimensi

```
pca = PCA()
pca.fit(X_train)

explained_variance_ratio = np.cumsum(pca.explained_variance_ratio_)
```

2.3. PCA dengan 10 Components

```
pca_10 = PCA(n_components=10)
X_train_pca_10 = pca_10.fit_transform(X_train)
X_test_pca_10 = pca_10.transform(X_test)
```

2.4. PCA dengan 30 Components

```
pca_30 = PCA(n_components=30)
X_train_pca_30 = pca_30.fit_transform(X_train)
X_test_pca_30 = pca_30.transform(X_test)
```

2.5. PCA dengan 50 Components

```
pca_50 = PCA(n_components=50)
X_train_pca_50 = pca_50.fit_transform(X_train)
X_test_pca_50 = pca_50.transform(X_test)
```

2.6. Latih Model Pada 10 Components

```
log_reg_10 = LogisticRegression(max_iter=1000, random_state=42)
log_reg_10.fit(X_train_pca_10, y_train)

y_pred_10 = log_reg_10.predict(X_test_pca_10)
accuracy_10 = accuracy_score(y_test, y_pred_10)
print(f"Akurasi dengan 10 component: {accuracy_10}")
```

```
Akurasi dengan 10 component: 0.8035714285714286
```

2.7. Latih Model pada 30 Component

```
log_reg_30 = LogisticRegression(max_iter=1000, random_state=42)
log_reg_30.fit(X_train_pca_30, y_train)

y_pred_30 = log_reg_30.predict(X_test_pca_30)
accuracy_30 = accuracy_score(y_test, y_pred_30)
print(f"Akurasi dengan 30 component: {accuracy_30}")
```

```
Akurasi dengan 30 component: 0.8925
```

2.8. Latih Model pada 50 Component

```
log_reg_50 = LogisticRegression(max_iter=1000, random_state=42)
log_reg_50.fit(X_train_pca_50, y_train)

y_pred_50 = log_reg_50.predict(X_test_pca_50)
accuracy_50 = accuracy_score(y_test, y_pred_50)
print(f"Akurasi dengan 50 component: {accuracy_50}")
```

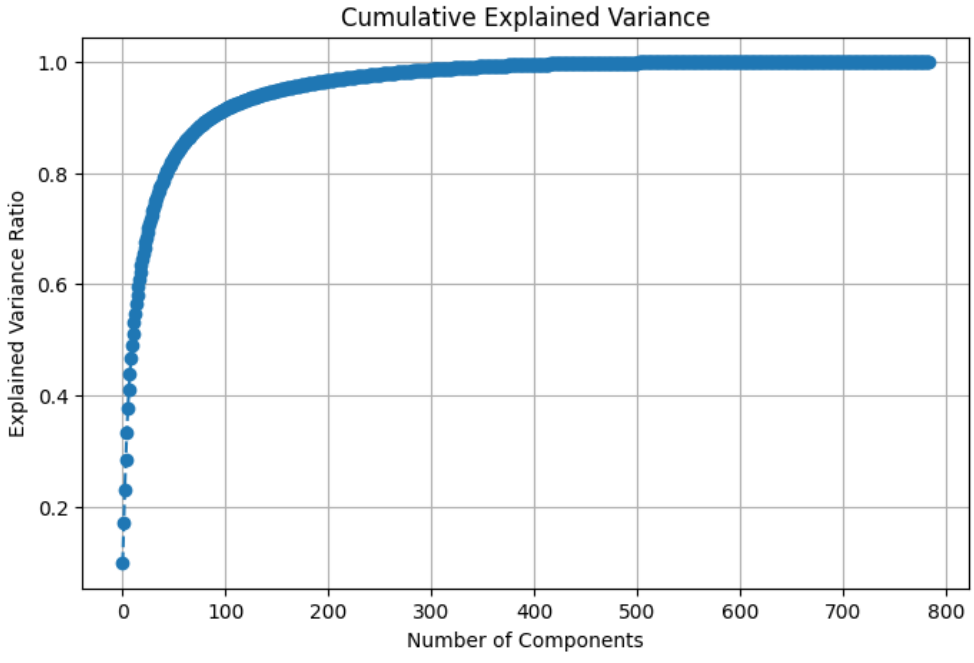
```
Akurasi dengan 50 component: 0.9082857142857143
```

2.9. Bandingkan Model Akurasi

```
# Compare Accuracies
print(f"Akurasi dengan 10 component: {accuracy_10}")
print(f"Akurasi dengan 30 component: {accuracy_30}")
print(f"Akurasi dengan 50 component: {accuracy_50}")
```

```
Akurasi dengan 10 component: 0.8035714285714286
Akurasi dengan 30 component: 0.8925
Akurasi dengan 50 component: 0.9082857142857143
```

2.10. Visualisasi Total Variansi



Hasil:

Untuk menjaga keseimbangan antara akurasi dan efisiensi, 30 komponen adalah pilihan yang tepat karena memberikan peningkatan akurasi yang signifikan dari 10 komponen (+8.89%), memiliki kompleksitas yang lebih ringan dibandingkan 50 komponen, namun hanya kehilangan sedikit akurasi (sekitar 1.6%).